SANGE -Stochastic Automata Networks Generator. A tool to efficiently predict events through structured Markovian models (extended version)

Joaquim Assunção, Paulo Fernandes, Lucelene Lopes, Angelika Studeny, Jean-Marc Vincent

SANGE - Stochastic Automata Networks Generator. A tool to efficiently predict events through structured Markovian models (extended version)

Joaquim Assunção, Paulo Fernandes , Lucelene Lopes , Angelika Studeny , Jean-Marc Vincent

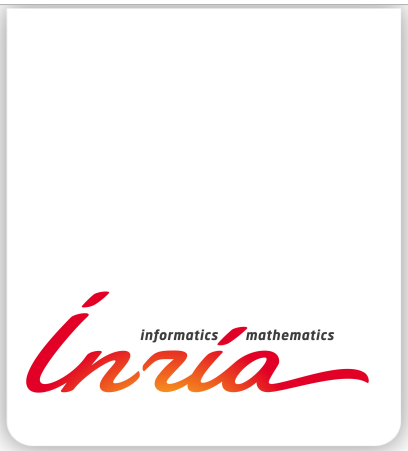
RESEARCH REPORT

N° 8724

May 2015

Project-Teams MESCAL

ISSN 0249-6399 ISRN INRIA/RR--8724--FR+ENG



SANGE - Stochastic Automata Networks Generator. A tool to efficiently predict events through structured Markovian models (extended version)

Joaquim Assunção *†, Paulo Fernandes †
, Lucelene Lopes †, Angelika Studeny *, Jean-Marc Vincent *

Project-Teams MESCAL

Research Report n° 8724 — May 2015 — 13 pages

Abstract: The use of stochastic formalisms, such as Stochastic Automata Networks (SAN), can be very useful for statistical prediction and behavior analysis. Once well fitted, such formalisms can generate probabilities about a target reality. These probabilities can be seen as a statistical approach of knowledge discovery. However, the building process of models for real world problems is time consuming even for experienced modelers. Furthermore, it is often necessary to be a domain specialist to create a model. This work illustrates a new method to automatically learn simple SAN models directly from a data source. This method is encapsulated in a tool called SAN GEnerator (SANGE). Through examples we show how this new model fitting method is powerful and relatively easy to use; therefore this can grant access to a much broader community to such powerful modeling formalisms.

Key-words: Stochastic Automata Networks, model fitting, time series prediction.

* {joaquim.assuncao, jean-marc.vincent, angelika.studeny}@inria.fr
† {joaquim.assuncao, paulo.fernandes, lucelene.lopes}@pucrs.br. PUCRS, Porto Alegre, Brazil

RESEARCH CENTRE
GRENOBLE – RHÔNE-ALPES

Inovallée
655 avenue de l'Europe Montbonnot
38334 Saint Ismier Cedex

SANGE - Générateur de réseaux d'automates stochastiques. Un outil efficace de prédiction d'événements grâce à des modèles markoviens structurés (version étendue)

Résumé : L'utilisation de formalismes structurés tels que les réseaux d'automates stochastiques permet, une fois la phase de calibration réalisée, de prédire le comportement de systèmes complexes à partir de leur distribution stationnaire. Toutefois, le processus de construction du modèle est complexe et coûteux en temps, et il demande souvent l'intervention d'un expert du domaine. Cet article présente une nouvelle méthode d'apprentissage, qui, automatiquement, construit un modèle sous forme de réseau d'automates stochastiques directement à partir des données (séries chronologiques). Cette méthode a été implantée dans un outil logiciel SAN GEnerator (SANGE). Sur quelques exemples on montre la simplicité d'utilisation de l'approche et la qualité de l'adéquation du modèle résultant aux données.

Mots-clés : Réseaux d'automates stochastiques, adéquation de modèle, prédiction de séries chronologiques.

1 Introduction

Stochastic Automata Networks (SAN) is a powerful formalism to describe systems as stochastic models. Through these models we can derive probabilities concerning some event or set of events of a system. Our research group has a record of successful development of stochastic models for behavior prediction of realities from several domains, *e.g.*, geological events [2], production lines [8], distributed software development teams [15] and mobility patterns [11]. In all these examples, the model construction required domain specialists and a large amount of stochastic modeling knowledge. The resulting models are very accurate in predicting the behavior of the realities as could be verified by comparison with records of each reality behavior.

Typically, SAN model construction is a top-down driven approach, *i.e.*, first the target reality is analyzed, then its behavior is translated into a stochastic model. Once we have a complete SAN model, it is possible to use a collection of specialized algorithms that can solve it [5]. The problem of this approach is that it is specific to a given system. In other words, each new system must be carefully analyzed before the creation of the model. This analysis usually is performed via handmade steps such as data analysis and data selection.

In a previous work [1] we proposed a bottom-up process to forecast events using time series and stochastic models (Fig 1). This modeling approach also speeds up the time to develop a representative model from input data. However, we did not have a technique to automatically generate SAN code, but only plain Markov chains (MC). The extension of this previous work to generate SAN models increases our potential to handle more complex (multidimensional) data.

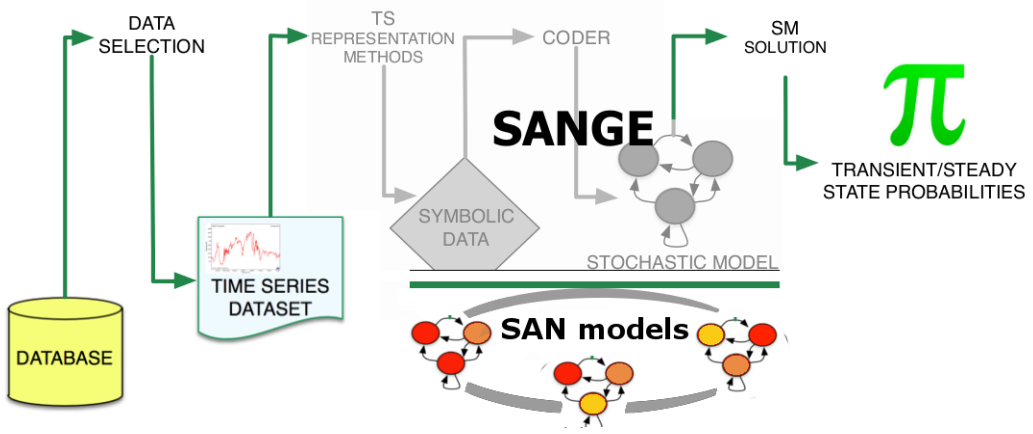


Figure 1: A Dimensionality Reduction Process to Forecast Events Through Stochastic Models [1] enhanced by the generation of SAN models (SANGE).

A bottom-up, MC-based, approach gives a solution for any generic model. However a plain (unstructured) MC is usually a limited, memory expensive model. Limited in the sense that you have a bulk representation for a system, regardless of how complex it may be. Memory expensive because a system with S states is represented by a transition probability matrix of S^2. In the best case, the number of non-zero entries will be in the order of $2S$.

SAN formalism is modular and its model representation is composed by a collection of sub-systems, which are usually much more compact benefiting from tensor representations. Thus, in this paper we show a new approach to fit SAN models directly to data, reducing the model size. This approach has been implemented in a tool called SANGE (SAN GEnerator) that automatically generates SAN models from a dataset.

The excessive human effort to construct models can be avoided with a tool that handles the formal tasks to convert data into state transitions. Consequently, the user can focus on more interesting tasks, such as interpreting the results and applying the gain knowledge. Additionally, SANGE performs dimensionality reduction using time series representation methods [12]. Thus, SANGE is also capable of automatically fitting the model to input data, possibly achieving better models than those made by humans.

Our algorithm was inspired by a well known and broadly used formalism, Hidden Markovian Models (HMM), and its fitting algorithm (Baum-Welch, BW) [3]. Thus, the HMM user only needs to understand the modeling basics and how to interpret the generated HMM model.

The BW algorithm is a special case of the Expectation-Maximization (EM) algorithm, using forward-backward probabilities to estimate the model parameters. Our approach can be seen as an adaptation of this algorithm implementing only the forward procedures. However, our solution works with a structured formalism, which naturally can provide higher accuracy and can be more user-friendly to describe a system.

This report is organized as follows: The next Sections briefly describe MCs, HMMs and SAN formalisms. Section 4 shows the techniques used to generate SAN models and how SANGE works internally. Finally, examples of the application are given with final remarks.

2 Markov chains and HMM

Stochastic modeling can be seen as the art of representing the behavior of a system by describing its possible states (discrete state space) and a transition function that establishes a stochastic process to represent the succession of states. Among such formalisms, one of the most common are the Markov chains [17]. Given an initial state, even a simple Markov chain can be used to forecast future states by computing probabilities for each possible state considering some amount of time.

Being a classic formalism, Markov chains are easy to handle and widely employed to perform quantitative analysis of virtually any reality. Unfortunately, complex systems tend to generate huge models that are hard to handle using ordinary Markov chains. Such problem is known as the state space explosion problem. To cope with this problem, it is wise to recur to structured Markovian formalisms, as the Performance Evaluation Process Algebras [9], or the Stochastic Automata Networks (SAN) [14]. These formalisms are as powerful as the Markov chains, but much easier to handle since they are naturally modular.

An extension of basic Markov chains are the so-called Hidden Markovian Models (HMM). These models apply Markov chains to described the stochastic evolution of an unobserved part of the dynamic system. These hidden states determines the probabilistic behavior of the observable process. HMM models can generate a set of probabilities through indirect emissions, *i.e.* the probability matrix is hidden and its goal is to generate the right conditions for the visible emissions.

Broadly used, one of the reasons that the HMM formalism is successful is the Baum-Welch algorithm. With a few parameters this algorithm is capable to receive a dataset and generate a fitted model that best describes the behavior of the data. Thus, by learning the patters among data, this model can generate probabilities and predictions concerning other sequences of data. In an unpretentiously way, we can say that the basic idea is a statistical approach of some machine learning algorithms.

Improving our previous work, now we are focused on bring the best way to make easier and most effectively the generation of SAN models. As the Baum-Welch algorithm do for HMM, our techniques can automatically create a fitted model according to the input data through the analysis of the system behavior.

3 Stochastic Automata Networks

SAN [14] is a formalism that describes a complete system as a collection of subsystems that interact with each other in a Markovian behavior. The main advantage of SAN is that we can represent large and complex systems in a human understandable model that is often faster to solve than a flat MC model.

Formally, a SAN model represents a system composed of N subsystems described each by a stochastic automata. The global state of a SAN model is a composition of the local state of each automaton, and the movement from one global state to another is triggered by events that can either change the local state of one single automaton (a local event), or change the state of multiple automata (a synchronizing event). Additionally, the rate of any event can represent one single stochastic process (a constant rate) or multiple stochastic processes chosen according to current automata local states (a functional rate).

The Markov chain equivalent to a SAN model is simply the composition of all automata and events, *i.e.*, the set of all global states and the transitions made possible by the occurrence of all events. However, it is never necessary to bring a SAN model to its equivalent Markov chain, since every SAN model has a memory efficient tensor internal representation that also enables the use of specialized algorithms [7] which are usually more efficient than those used for solving large Markov chains. Comparing to HMMs that are usually forced to have small models [20], *i.e.* small number of transition states, SAN have have the potential to be more accurate without loose performance. More details about the SAN formalism can be obtained in [6].

4 Computational Kernel

SANGE's main objective is to reduce the time spent to generate stochastic models, thus, making possible the use of stochastic models by non-specialists. However, our solution needs records of a system behavior in the form of time series that will be assigned to the variables of interest for the system.

SANGE's basic operation consists in the composition of a set of time series describing how system variables behave [12]. Considering a system with n interest variables ($v^{(i)}$ with $i = 1,\ldots,n,$) we need a behavior sample of the system in the form of n time series with the successive values of the variables through time. With these time series, the basic process is to identify the points of interest in time, as the time ticks where at least one of the variables change its value. Once the time ticks of interest are identified, it is necessary to identify the possible values for the variables in order to define the stochastic model. This process is summarized in an example with three time series in Fig. 2 and Fig. 3 showing the identification of 11 time ticks (a) and the three succession of values for variables $v^{(1)}$, $v^{(2)}$ and $v^{(3)}$ as shown in Table 1:

$v^{(1)} \rightarrow$	$v_3^{(1)}$	$v_3^{(1)}$	$v_3^{(1)}$	$v_5^{(1)}$	$v_5^{(1)}$	$v_5^{(1)}$	$v_5^{(1)}$	$v_1^{(1)}$	$v_4^{(1)}$	$v_4^{(1)}$	$v_2^{(1)}$	$v_3^{(1)}$
$v^{(2)} \rightarrow$	$v_1^{(2)}$	$v_1^{(2)}$	$v_1^{(2)}$	$v_1^{(2)}$	$v_2^{(2)}$	$v_2^{(2)}$	$v_1^{(2)}$	$v_1^{(2)}$	$v_3^{(2)}$	$v_3^{(2)}$	$v_4^{(2)}$	$v_1^{(2)}$
$v^{(3)} \rightarrow$	$v_4^{(3)}$	$v_1^{(3)}$	$v_3^{(3)}$	$v_3^{(3)}$	$v_3^{(3)}$	$v_1^{(3)}$	$v_4^{(3)}$	$v_4^{(3)}$	$v_2^{(3)}$	$v_1^{(3)}$	$v_1^{(3)}$	$v_4^{(3)}$

Table 1: Sequence of values for variables $v^{(1)}$, $v^{(1)}$ and $v^{(3)}$

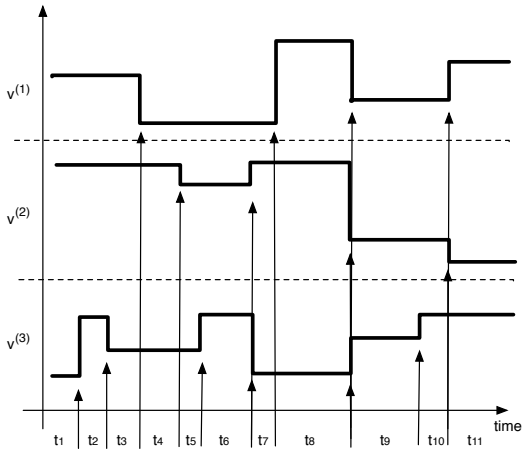


Figure 2: Example of SANGE basic process to three time series - identifying time ticks of interest

Fig. 2 and Fig. 3 examples would result in three automata to represent as local states each variable possible value, and transitions representing each state change through events. Fig. 4 presents the SAN model for this example.

To compute the rates of each local event we must take into consideration how many possible transitions are possible from each each origin local state. For example, event e_1 goes from state $v_3^{(1)}$ to state $v_5^{(1)}$, and observing all time ticks we see state $v_3^{(1)}$ as origin at the end of time ticks t_1, t_2 and t_3. In time ticks t_1 and t_2 automaton $V^{(1)}$ does not change state, and in t_3 it goes from $v_3^{(1)}$ to $v_5^{(1)}$. Therefore, one in three times event occurs, and rate of event e_1 becomes $1/3$.

For synchronizing events the computation of event rate is similar, but since more than one automaton

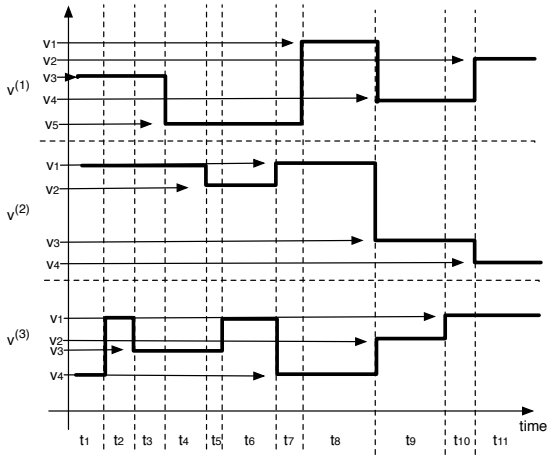


Figure 3: Example of SANGE basic process to three time series - identifying variable values succession

type	event	rate	type	event	rate	type	event	rate
loc	e_1	0.33	loc	e_5	0.33	syn	s_1	1.00
loc	e_2	0.25	loc	e_6	0.33	syn	s_2	1.00
loc	e_3	0.16	loc	e_7	1.00	syn	s_3	0.50
loc	e_4	0.25				syn	s_4	1.00

Table 2: SAN configuration generated

is concerned, is necessary to take into account the number of times the origin combination of states occurs. For example, event s_3 concerns automata $V^{(1)}$ and $V^{(2)}$ that start from combination of states $v_4^{(1)}$ and $v_3^{(2)}$, a combination that occurs at the end of time ticks t_9 and t_{10}, but event s_3 only happens in one of these two situations (after t_{10}), hence, event s_3 rate is $1/2$.

From a practical point of view the current version of SANGE can generate either a SAN model or a Markov chain output (which is basically a SAN model with a single automaton and only local events). The output physical format is a `.san` file, that can be accepted as input for a SAN solver like PEPS [5] that performs state of art Kronecker solutions [6].

Obviously, the presented example has quite small time series and it limits considerably the validity of the generated model. In real cases, much larger time series must be considered in order to see patterns that will be very important to bring statistical relevance to consider a larger number of states succession representative of an event. Naturally, in the stochastic related works, large amounts of samples are crucial.

However, it is easy to generate time series for most of real life systems. In fact, the literature states as more difficult to derive from real systems observed data the symbolic values of each time series [19, 18, 13, 10]. In our example, we already started with the time series dully coded in symbolic values (v_1, v_2, *etc.*), but several data can be obtained with continuous values, or even highly dimensional categories. Even though such symbolic coding may affect the quality of the produced model, there are quite efficient symbolic coders available [12].

Currently, SANGE can handle two formalisms; Markov Chains (MC) and Stochastic Automata Networks (SAN). Despite the many available solutions for Markov chains, generate *.san* code is a breakthrough that can speed up the process of modeling large systems. Thus, we can generate and solve millions of states in a few seconds. In SANGE, the technique to generate *.san* code share some concepts with the generation of Markov chain. Thus, this section explains the input format, creation and execution

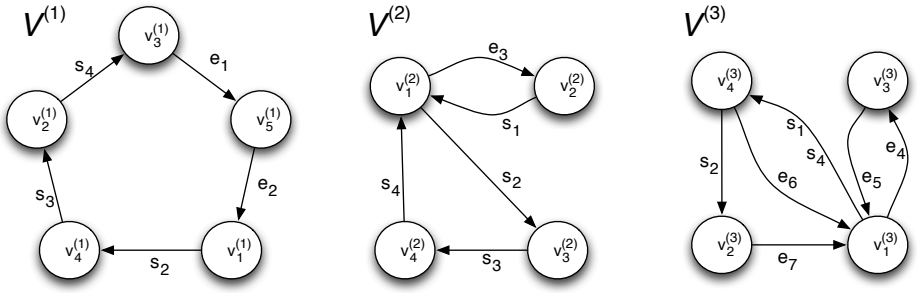


Figure 4: Equivalent SAN model for the three time series example of Fig. 3.

of Markov Chains aiming to describe the whole tool and its primary utility, generate *.san* code.

5 Example 1: Weather in Gothan city

To facilitate the understanding, we use an example commonly used in Markovian models, weather forecast [17][16][20]. The most basic example are given by a Markov chain with 3 states, representing the states **R**aining, **S**unny and **C**loudy (Figure 5). These states have assigned transactions that represents the probabilities from a given state go to another, or stay in the same.

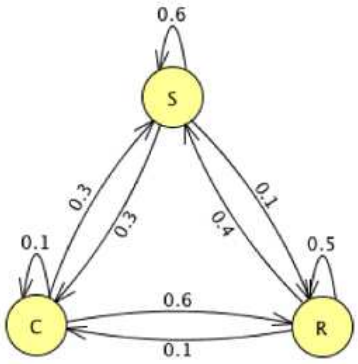


Figure 5: Classical example Markov chain model.

Solving this model we can achieve the transient and stationary probabilities of being in one of the three states. However, this is directly represented by a probability matrix, which makes unpractical for real world applications. Furthermore, in the most known formalisms, Markov chains and Hidden Markov chains, it is not possible to make a model with multiple automata in a unique system, *i.e.* they do not have a structure for this. Using SAN with SANGE is possible to generate a structured version, which allows us to assemble more elements to our model and solve those as one.

SANGE encapsulates our solution to bring these basics and more advanced statistical techniques. As previously described, we merge the SAN practical characteristics with the practicality of TS. Thus, the following example is solved using the same techniques described in the previous Section 4. However, it is important to emphasize that the following example do not consider the reality of data neither the reality of the model; Our goal is to illustrate how SANGE works with multiple variables and how easy is

to create a model through the use of SANGE. Although SANGE is a prototype, the basic functions are implemented and some basic models can be generated.

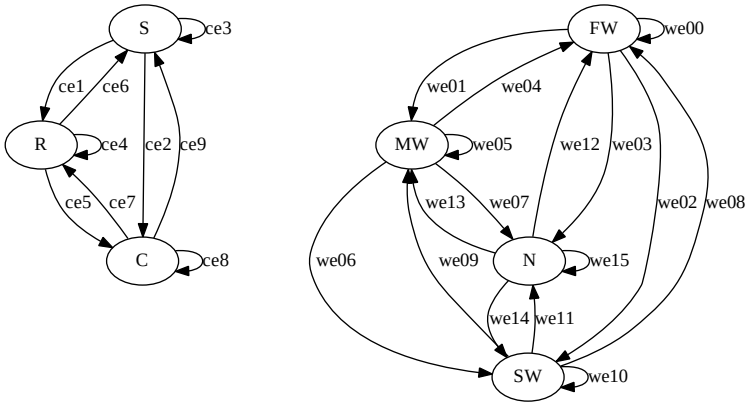


Figure 6: Generic SAN model for weather conditions and wind force.

In the Figure 6, for ease of understanding, we show a generic model that have 2 automata, one for weather conditions and other for the wind's force. **'ce'** stands for climate event and **'we'** for wind event. The state's labels in the first automaton means **R**aining, **S**unny and **C**loudy. In the second automaton, the states are related to the wind velocity, **F**ast, **M**edium, **S**low and **N**one.

Assuming that this model is accurate to predict the wind and the climate of a city called *Gothan*, and that we want to know the probability of Gothan face a rain and fast wind at the same time. We do not have many records, so we need to discover by a very small sample formatted as the following table:

Table 3: Sample for input data

Raw data		Symbolic data	
weather	**wind speed**	**weather**	**wind speed**
cloudy	48	b	d
raining	16	c	b
sunny	26	a	c
...	...	...	...
sunny	9	a	a

In this case, the probability to have rain and fast wind is 4.5%. In this example, a basic combination of 2 automata was used; however the number of automata and states can be much bigger. Figure 4 shows that the maximum number of represented states shall be 80 states. This is a very compact representation and through SANGE, it scales better and demands less effort This is a very compact representation and through SANGE; it scales better and demands less effort than the equivalent Markov model, allowing an easier derivation of probabilities in large models.

With SANGE, there is a demonstration file containing these data. After fulfill the very basic parameters, all that must be done is load the data through the interface and the model shall be generated.

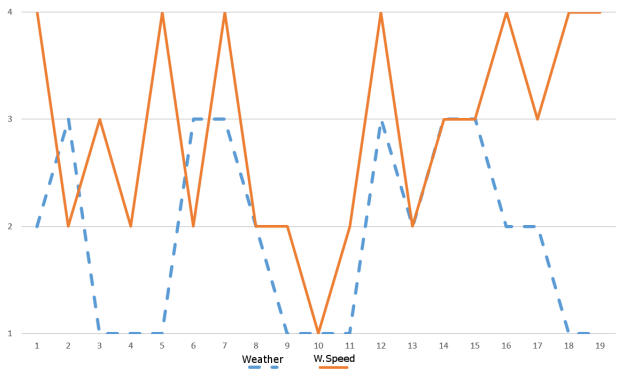


Figure 7: Line plot after the symbolic representation. Each letter is assigned to a value, $a = 1$, $b = 2$, $c = 3$, $d = 4$.

6 Example 2: Father-son height

This section demonstrates the use of SANGE and its results through the use of a dataset that stores the height of fathers and sons [1]. Our goal is to demonstrate the use of SANGE with a real dataset. Unlike the previous example, this dataset have 1078 records that not follows a time constraint; thus, a non-temporal data. Furthermore, we highlight the importance of an automatically generation, which allows a common user to use SAN models with higher amounts of data.

This dataset is composed by two variables, Father's and Son's heights, in inches. Figure 8 shows the relation between father's and son's height. The center line represents the linear prediction, delimited by the upper and lower bounds (red lines). Through this model, given a father's height, we can know the expected height of a son.

However, there are probabilistic questions that are not directly answered by these kind of models. *E.g.*., What is the probability to a short father, between $1.63m$ and $1.64m$, have a son with a height between $1.70m$ and $1.72m$? The mark ⊕ on the Fig. 8 is in the center of this region.

Despite do not have a temporal constraint, this relation still can be modeled or automatically fitted by SANGE. Directly from a dataset we can get classes for these values, *e.g.*, 63.2 to 63.9 inches. These classes can be controlled by the number of variables, it generates a set of symbols (A PAA like technique [12]) that will become states in a SAN automaton. Then, these states will be linked according to the techniques described in Section 4. Finally, SANGE generates a SAN file that can be processed by a SAN algorithm [5].

For this example, running the SAN file, we achieve 1.5%. Although this simple example can be easily modeled, SANGE instantaneously generates the necessary code to derive the solution. Furthermore, it is possible to achieve solutions for similar problems with more variables and more data. These variables can be replaced by others that can represent nearly any other problem. In fact, in the final version of the tool, we will be restricted only by the total number of states generated; which can be up to millions in an ordinary personal computer, and still be solved in seconds [4]. Furthermore, the dimensionality reduction in this fitting method 4 enable us to generate small automata with large number of records in by variable.

[1] This data is public available in the R package "UsingR". Also, all the files can be directly downloaded at `www.joaquim.pro.br/seke2015`

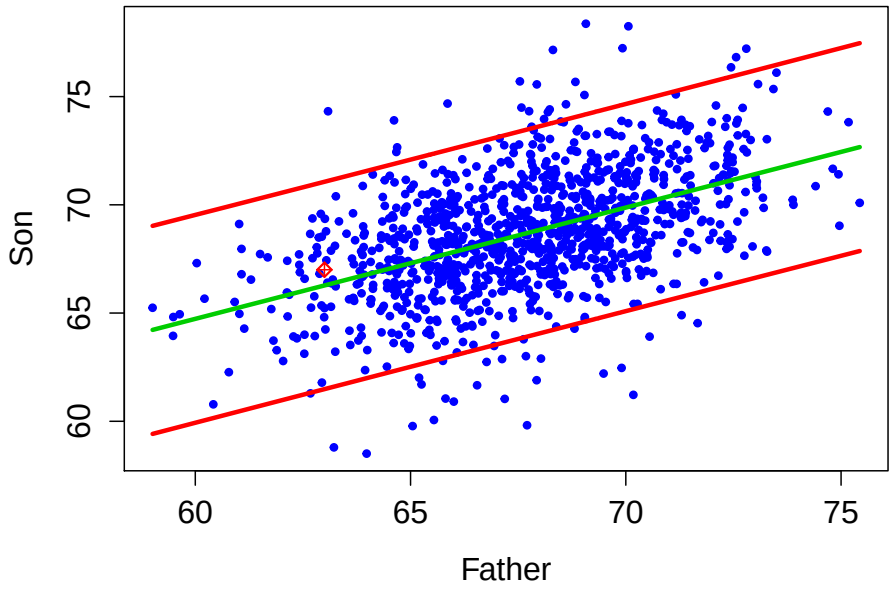


Figure 8: Linear model showing the relation between Father's height and son's height (In inches).

7 Final Remarks

As the core technique to data mining, statistics are important to knowledge discovery. It allows us to infer probabilities through samples instead of considering the complete behavior data. Stochastic formalisms are heavily based on probabilistic techniques. The use of stochastic modeling tools is promising to forecast the behavior of systems which can be described as sets of time series.

Our main achievement was to introduce SAN formalism to the process (illustrated in Fig.1) in an automated way, allowing non specialist users to take advantage of SAN's structure and solutions. Through the automatic fitting, we create a bottom-up approach that can be broadly used, once that the learning process avoids the human effort to manually create such models. If compared to the traditional modeling approach, our implemented solution, SANGE, represents an interesting option that simplifies the effort to construct SAN models. Thus, considering that SANGE is a first attempt to automatically generate SAN models that can be useful to real world datasets, we bring a new method for knowledge discovery through stochastic modeling.

We presented a first version of SANGE that is already functional and it allows further studies about the benefits of this forecasting approach. As future work, we will improve our algorithm by adapting the forward-backward procedures from BW algorithm probabilities. Then, through a detailed comparison of the computational cost we can compare the performance of both formalisms.

For future work, we will improve our algorithm by adapting the forward-backward procedures from BW algorithm, improving SANGE capacity to handle more complex SAN models for real datasets.

8 Acknowledgments

“This work was conducted during a scholarship supported by the International Cooperation Program CAPES/COFECUB at the University Joseph Fourier. Financed by CAPES - Brazilian Federal Agency for Support and Evaluation of Graduate Education within the Ministry of Education of Brazil."

References

[1] J. Assunção, P. Fernandes, L. Lopes, and S. Normey. A dimensionality reduction process to forecast events through stochastic models. In *The 26th International Conference on Software Engineering & Knowledge Engineering, SEKE 2014*, pages 534–539, HyattRegencyHotel, Vancouver, Canada, Jul 2014. Knowledge Systems Institute Graduate School. ISBN-13: 978-1-891706-35-7.

[2] J. Assunção, L. Espindola, P. Fernandes, M. Pivel, and A. Sales. A structured stochastic model for prediction of geological stratal stacking patterns. In *6th Practical Applications of Stochastic Modelling (PASM'12)*, pages 1–15, London, UK, September 2012.

[3] L. E. Baum, T. Petrie, G. Soules, and N. Weiss. A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains. *The Annals of Mathematical Statistics*, 41(1):164–171, 02 1970.

[4] L. Brenner, P. Fernandes, B. Plateau, and I. Sbeity. PEPS 2007 - Stochastic Automata Networks Software Tool. In *International Conference on Quantitative Evaluation of Systems (QEST 2007)*, pages 163–164. IEEE Press, September 2007.

[5] L. Brenner, P. Fernandes, B. Plateau, and I. Sbeity. PEPS2007 - Stochastic Automata Networks Software Tool. In *Proceedings of the 4th International Conference on Quantitative Evaluation of SysTems (QEST 2007)*, pages 163–164, Edinburgh, UK, September 2007. IEEE Computer Society.

[6] L. Brenner, P. Fernandes, and A. Sales. The Need for and the Advantages of Generalized Tensor Algebra for Kronecker Structured Representations. *International Journal of Simulation: Systems, Science & Technology (IJSIM)*, 6(3-4):52–60, February 2005.

[7] R. M. Czekster, P. Fernandes, and T. Webber. Efficient vector-descriptor product exploiting time-memory trade-offs. *ACM SIGMETRICS Performance Evaluation Review*, 39(3):2–9, December 2011. doi: 10.1145/2160803.2160805.

[8] P. Fernandes, M. O'Kelly, C. Papadopoulos, and A. Sales. Analysis of exponential reliable production lines using kronecker descriptors. *Int. Journal of Production Research*, 51(9):2511–2528, 2013.

[9] J. Hillston. *A compositional approach to performance modelling*. Cambridge University Press, New York, USA, 1996.

[10] Keogh, Eamonn and Wei, Li and Xi, Xiaopeng and Lee, Sang-Hee and Vlachos, Michail. Lb-keogh supports exact indexing of shapes under rotation invariance with arbitrary representations and distance measures. In *Proceedings of the 32nd international conference on Very large data bases*, VLDB '06, pages 882–893. VLDB Endowment, 2006.

[11] F. L. Dotti, P. Fernandes, and C. Nunes. Structured Markovian models for discrete spatial mobile node distribution. *Journal of the Brazilian Computer Society*, 17:31–52, 2011. doi: 10.1007/s13173-010-0026.

[12] J. Lin, E. Keogh, S. Lonardi, and B. Chiu. A symbolic representation of time series, with implications for streaming algorithms. In *Proceedings of the 8th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery*, DMKD '03, pages 2–11, New York, NY, USA, 2003. ACM.

[13] J. Peralta, G. Gutierrez, and A. Sanchis. Shuffle design to improve time series forecasting accuracy. In *Evolutionary Computation, 2009. CEC '09. IEEE Congress on*, pages 741–748, 2009.

[14] B. Plateau. On the stochastic structure of parallelism and synchronization models for distributed algorithms. *ACM SIGMETRICS Performance Evaluation Review*, 13(2):147–154, August 1985.

[15] A. Santos, A. Sales, and P. Fernandes. Setting up a stochastic model for teams working in a follow-the-sun environment. In *Global Software Engineering (ICGSE), 2012 IEEE Seventh International Conference on*, pages 179–179, 2012.

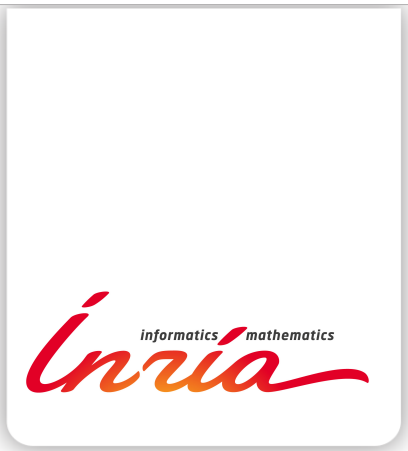
[16] W. Stewart. *Introduction to the numerical solution of Markov chains*. PRINCETON University Press, 1994.

[17] W. J. Stewart. *Probability, Markov Chains, Queues, and Simulation*. Princeton University Press, USA, 2009.

[18] Z. Tang, C. de Almeida, and P. A. Fishwick. Time series forecasting using neural networks vs. box- jenkins methodology. *SIMULATION*, 57(5):303–310, 1991.

[19] L. Ye and E. Keogh. Time series shapelets: a new primitive for data mining. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '09, pages 947–956, New York, NY, USA, 2009. ACM.

[20] W. Zucchini and I. MacDonald. *Hidden Markov Models for Time Series: An Introduction Using R*. Chapman & Hall/CRC Monographs on Statistics & Applied Probability. Taylor & Francis, 2009.



RESEARCH CENTRE
GRENOBLE – RHÔNE-ALPES

Inovallée
655 avenue de l'Europe Montbonnot
38334 Saint Ismier Cedex

Publisher
Inria
Domaine de Voluceau - Rocquencourt
BP 105 - 78153 Le Chesnay Cedex
inria.fr

ISSN 0249-6399